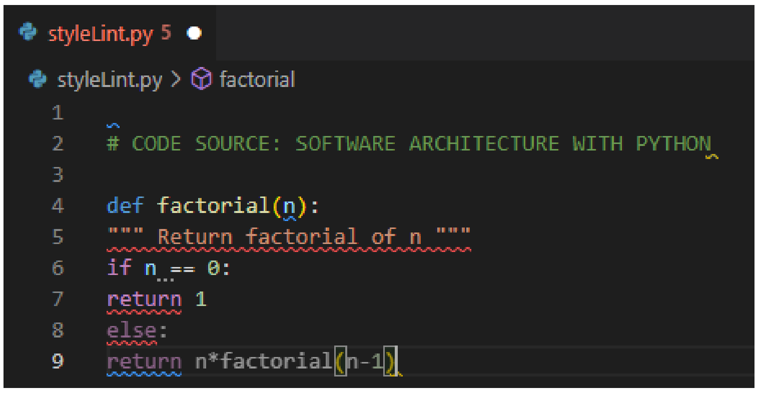
Linters

Tasks provided by University of Essex’s Codio Workspace on Testing with Python.

Task 1

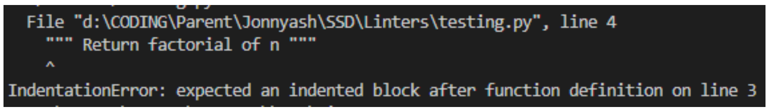
What happens when the code is run?

Input



```
styleLint.py 5 ●
styleLint.py > factorial

# CODE SOURCE: SOFTWARE ARCHITECTURE WITH PYTHON

def factorial(n):
""" Return factorial of n """
if n == 0:
return 1
else:
return n*factorial(n-1)
```

Output



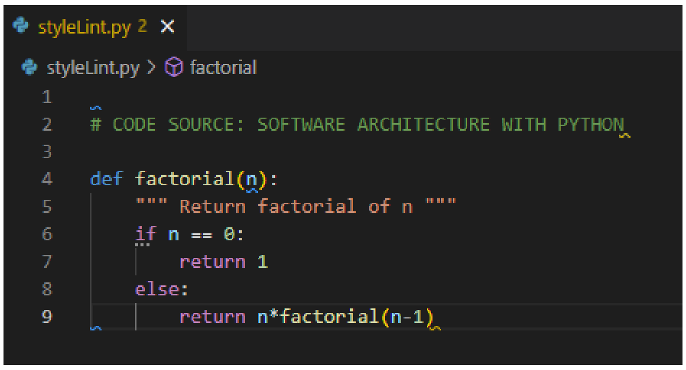
```
  File "d:\CODING\Parent\Jonnyash\SSD\Linters\testing.py", line 4
    """ Return factorial of n """
    ^
IndentationError: expected an indented block after function definition on line 3
```

Can you modify this code for a more favourable outcome?

Yes, formatting and indenting.

What amendments have you made to the code?

Input after alterations



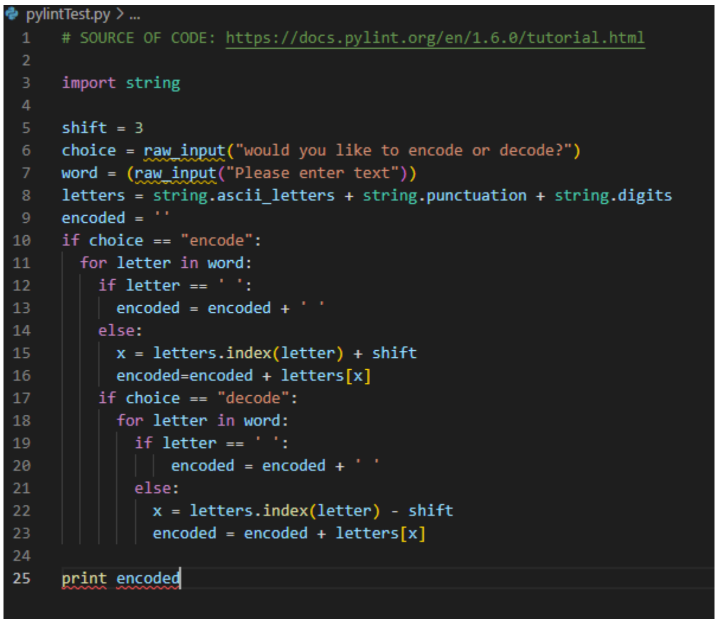
```python

# CODE SOURCE: SOFTWARE ARCHITECTURE WITH PYTHON

def factorial(n):
    """ Return factorial of n """
    if n == 0:
        return 1
    else:
        return n*factorial(n-1)
```

Task 2

Review each of the code errors returned.

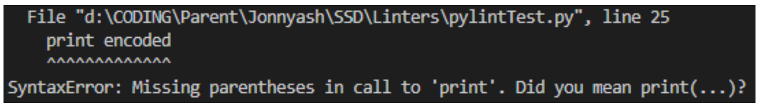
Can you correct each of the errors identified by pylint?

input



```python
# SOURCE OF CODE: https://docs.pylint.org/en/1.6.0/tutorial.html

import string

shift = 3
choice = raw_input("would you like to encode or decode?")
word = (raw_input("Please enter text"))
letters = string.ascii_letters + string.punctuation + string.digits
encoded = ''
if choice == "encode":
  for letter in word:
    if letter == ' ':
      encoded = encoded + ' '
    else:
      x = letters.index(letter) + shift
      encoded=encoded + letters[x]
    if choice == "decode":
      for letter in word:
        if letter == ' ':
          encoded = encoded + ' '
        else:
          x = letters.index(letter) - shift
          encoded = encoded + letters[x]

print encoded
```

Output

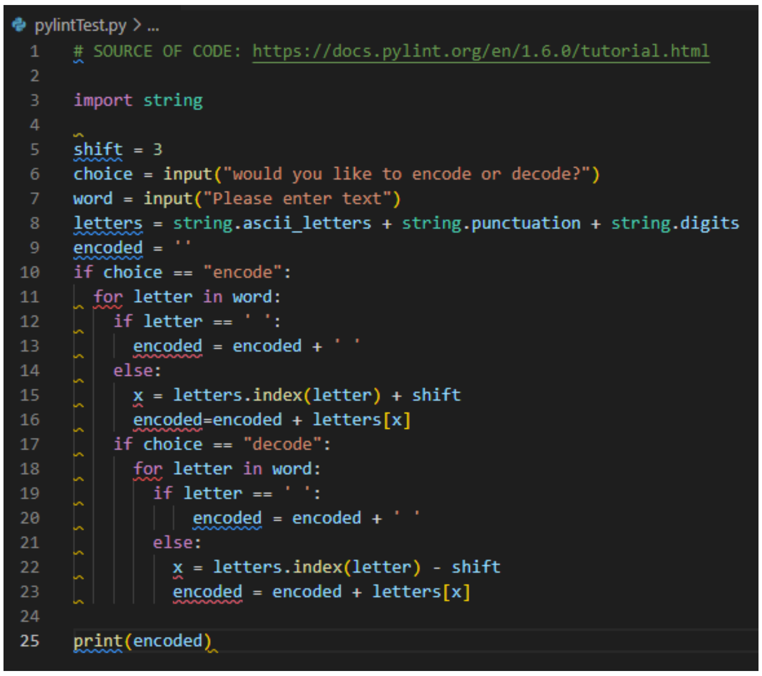


```
  File "d:\CODING\Parent\Jonnyash\SSD\Linters\pylintTest.py", line 25
    print encoded
    ^^^^^^^^^^^^^
SyntaxError: Missing parentheses in call to 'print'. Did you mean print(...)?
```

Can you correct each of the errors identified by pylint?

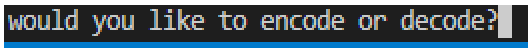
>Yes, deleted ‘raw_’, reduced whitespace between ‘print encoded’, and put parenthesis around ‘encdoded’.

Input



```
pylintTest.py > ...
# SOURCE OF CODE: https://docs.pylint.org/en/1.6.0/tutorial.html

import string

shift = 3
choice = input("would you like to encode or decode?")
word = input("Please enter text")
letters = string.ascii_letters + string.punctuation + string.digits
encoded = ''
if choice == "encode":
    for letter in word:
        if letter == ' ':
            encoded = encoded + ' '
        else:
            x = letters.index(letter) + shift
            encoded=encoded + letters[x]
        if choice == "decode":
            for letter in word:
                if letter == ' ':
                    encoded = encoded + ' '
                else:
                    x = letters.index(letter) - shift
                    encoded = encoded + letters[x]

print(encoded)
```

Output



```
would you like to encode or decode?
```

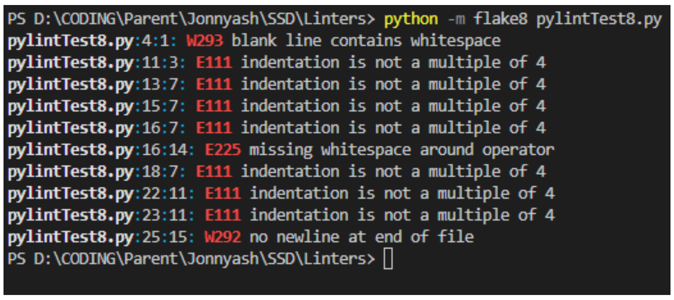
Task 3

Run flake8 on pylintTest.py

Review the errors returned. In what way does this error message differ from the error message returned by pylint?

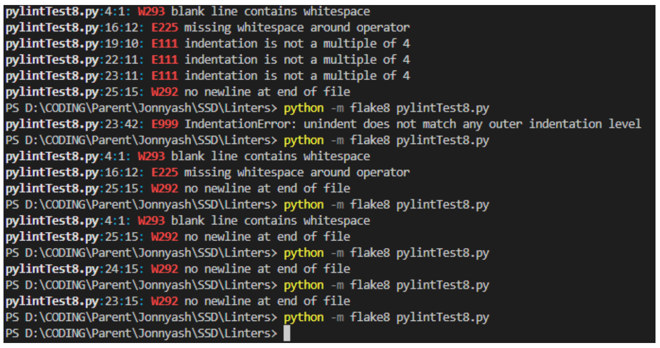
>I found using Flake8 easier than pylint as can be seen in the differences below.

Errors produced after running Flake8



```
PS D:\CODING\Parent\Jonnyash\SSD\Linters> python -m flake8 pylintTest8.py
pylintTest8.py:4:1: W293 blank line contains whitespace
pylintTest8.py:11:3: E111 indentation is not a multiple of 4
pylintTest8.py:13:7: E111 indentation is not a multiple of 4
pylintTest8.py:15:7: E111 indentation is not a multiple of 4
pylintTest8.py:16:7: E111 indentation is not a multiple of 4
pylintTest8.py:16:14: E225 missing whitespace around operator
pylintTest8.py:18:7: E111 indentation is not a multiple of 4
pylintTest8.py:22:11: E111 indentation is not a multiple of 4
pylintTest8.py:23:11: E111 indentation is not a multiple of 4
pylintTest8.py:25:15: W292 no newline at end of file
PS D:\CODING\Parent\Jonnyash\SSD\Linters>
```

No errors after fixing



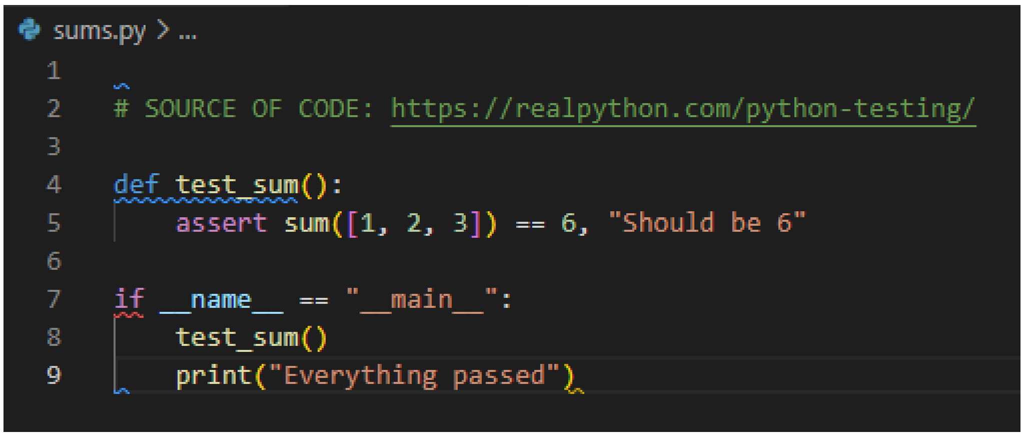
```
pylintTest8.py:4:1: W293 blank line contains whitespace
pylintTest8.py:16:12: E225 missing whitespace around operator
pylintTest8.py:19:10: E111 indentation is not a multiple of 4
pylintTest8.py:22:11: E111 indentation is not a multiple of 4
pylintTest8.py:23:11: E111 indentation is not a multiple of 4
pylintTest8.py:25:15: W292 no newline at end of file
PS D:\CODING\Parent\Jonnyash\SSD\Linters> python -m flake8 pylintTest8.py
pylintTest8.py:23:42: E999 IndentationError: unindent does not match any outer indentation level
PS D:\CODING\Parent\Jonnyash\SSD\Linters> python -m flake8 pylintTest8.py
pylintTest8.py:4:1: W293 blank line contains whitespace
pylintTest8.py:16:12: E225 missing whitespace around operator
pylintTest8.py:25:15: W292 no newline at end of file
PS D:\CODING\Parent\Jonnyash\SSD\Linters> python -m flake8 pylintTest8.py
pylintTest8.py:4:1: W293 blank line contains whitespace
pylintTest8.py:25:15: W292 no newline at end of file
PS D:\CODING\Parent\Jonnyash\SSD\Linters> python -m flake8 pylintTest8.py
pylintTest8.py:24:15: W292 no newline at end of file
PS D:\CODING\Parent\Jonnyash\SSD\Linters> python -m flake8 pylintTest8.py
pylintTest8.py:23:15: W292 no newline at end of file
PS D:\CODING\Parent\Jonnyash\SSD\Linters> python -m flake8 pylintTest8.py
PS D:\CODING\Parent\Jonnyash\SSD\Linters>
```

Task 3

Run mccabe on sums.py

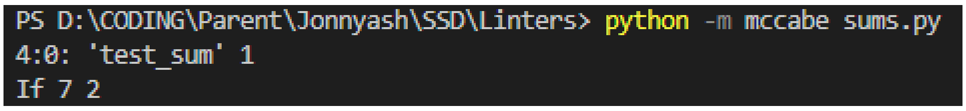
What is the result?

Input



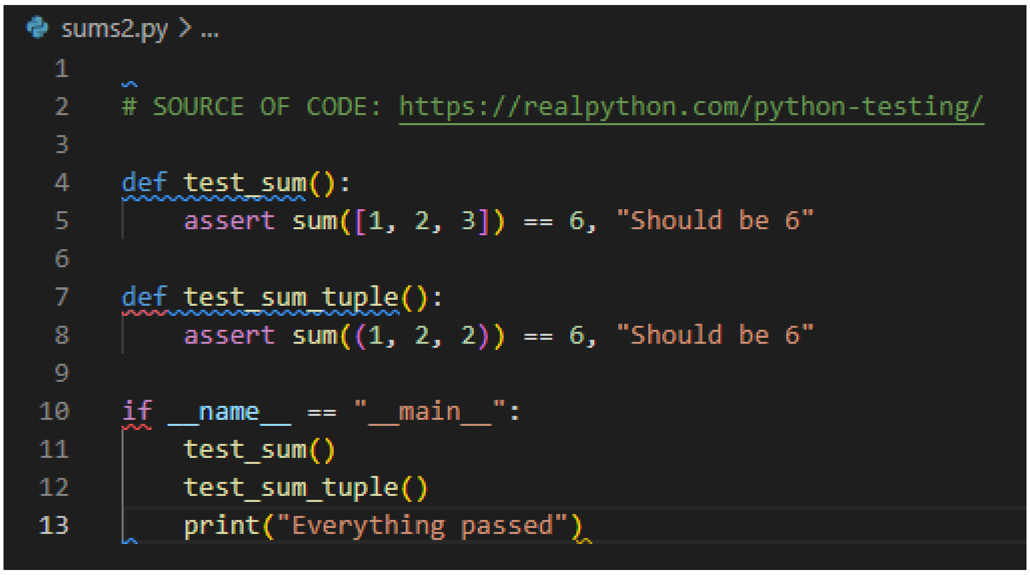
```
sums.py > ...

# SOURCE OF CODE: https://realpython.com/python-testing/

def test_sum():
    assert sum([1, 2, 3]) == 6, "Should be 6"

if __name__ == "__main__":
    test_sum()
    print("Everything passed")
```

Output



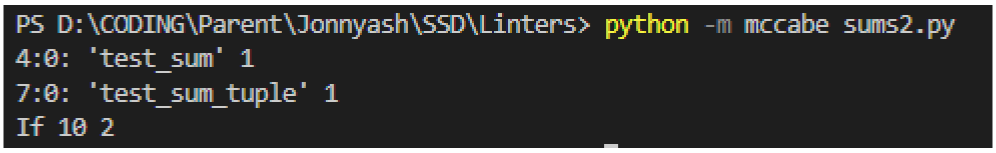
```
PS D:\CODING\Parent\Jonnyash\SSD\Linters> python -m mccabe sums.py
4:0: 'test_sum' 1
If 7 2
```

Run mccabe on sums2.py



```
sums2.py > ...

# SOURCE OF CODE: https://realpython.com/python-testing/

def test_sum():
    assert sum([1, 2, 3]) == 6, "Should be 6"

def test_sum_tuple():
    assert sum((1, 2, 2)) == 6, "Should be 6"

if __name__ == "__main__":
    test_sum()
    test_sum_tuple()
    print("Everything passed")
```

What is the result?



```
PS D:\CODING\Parent\Jonnyash\SSD\Linters> python -m mccabe sums2.py
4:0: 'test_sum' 1
7:0: 'test_sum_tuple' 1
If 10 2
```

What are the contributors to the cyclomatic complexity in each piece of code?

>The programs have conditionals and functions.

Task 4

Exploring the Cyclomatic Complexity's Relevance Today

The Cyclomatic Complexity is commonly considered in modules on testing the validity of code design today. However, in your opinion, should it be?

Does it remain relevant today?

Specific to the focus of this module, is it relevant in our quest to develop secure software?

Yes, it helps to better understand and reduce code complexity.

According to (2018), cyclomatic complexity has many advantages and disadvantages.

Advantages:

- Used as a quality metric- gives complexity relative to various designs.
- Faster than Halstead's metrics.
- Measures best areas of concentration and minimum effort for testing.
- Helps to guide testing process.
- Easy application.

Disadvantages:

- Measures program's control complexity but not the data complexity.
- Nested conditional structures are harder to understand than non-nested structures.
- May give misleading figures for simple comparisons and design structures.

References

GeeksforGeeks. (2018) *Cyclomatic Complexity - GeeksforGeeks*. Available from: https://www.geeksforgeeks.org/cyclomatic-complexity/ [Accessed 2 Sep. 2022].